PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2000-149048
(43)Date of publication of application : 30.05.2000

(51)Int.Cl. G06T 15/00

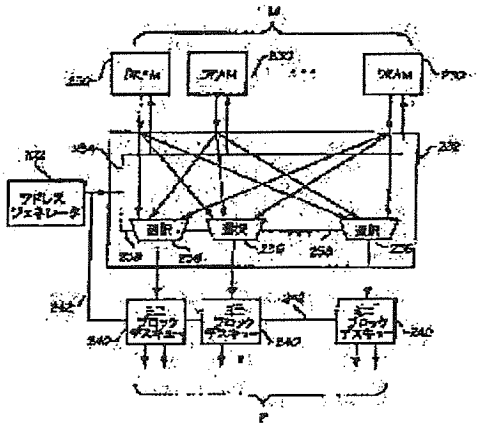
(21)Application number : 11-292817
(22)Date of filing : 14.10.1999

(71)Applicant : MITSUBISHI ELECTRIC INF TECHNOL CENTER AMERICA INC
(72)Inventor : KNITTEL JAMES
PEET WILLIAM
CORRELL KENNETH

(30)Priority
Priority number : 98 191865 Priority date : 12.11.1998 Priority country : US

(54) REAL-TIME VOLUME RENDERING ATTAINING METHOD

(57)Abstract:
PROBLEM TO BE SOLVED: To change fundamental architecture for the storage and distribution of voxels, in order to attain the performance of real-time volume rendering.
SOLUTION: Voxel data are stored in mini blocks which are allocated to a set of DRAM modules in a volume rendering system. Then a double-layered skewing architecture is assembled in a memory subsystem of the volume rendering system. Thus, the data can be transferred at the highest burst rate for a DRAM, and real-time volume rendering is attained. In each DRAM module, the mini blocks are allocated to the memory banks so that the miniblocks which are continuously accessed are allocated to the different banks so as to avoid idle cycles during the transfer of data and to cause the DRAM transfer efficiency to improve up to almost 100%.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号
特開2000－149048
(P2000－149048A)
(43)公開日　平成12年５月30日(2000.5.30)

(51)Int.Cl.'	識別記号	ＦＩ		テーマコード'(参考)
Ｇ０６Ｔ　15/00		Ｇ０６Ｆ　15/72	４５０Ｋ	

審査請求　有　請求項の数７　ＯＬ　（全 19 頁）

(21)出願番号　特願平11－292817

(22)出願日　平成11年10月14日(1999.10.14)

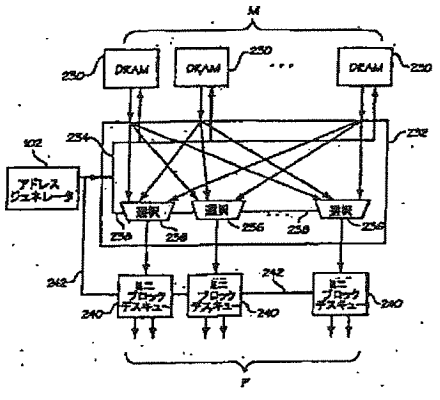
(31)優先権主張番号　０９／１９１８６５
(32)優先日　平成10年11月12日(1998.11.12)
(33)優先権主張国　米国（US）

(71)出願人　597067574
ミツビシ・エレクトリック・インフォメイション・テクノロジー・センター・アメリカ・インコーポレイテッド
MITSUBISHI ELECTRIC INFORMATION TECHNOLOGY CENTER AMERICA, INC.
アメリカ合衆国、マサチューセッツ州、ケンブリッジ、ブロードウエイ　201
(74)代理人　100057874
弁理士　曾我　道照　（外６名）

最終頁に続く

(54)【発明の名称】　実時間ボリュームレンダリングを可能にする方法

(57)【要約】
【課題】　実時間ボリュームレンダリングのパフォーマンスを達成するために、ボクセルの記憶および分散の根本的なアーキテクチャにおける変更を提供する。
【解決手段】　１組のＤＲＡＭメモリモジュールに割り当てられたミニブロック内にボクセルデータが記憶されたボリュームレンダリングシステムのメモリサブシステム上に２層のスキューイングアーキテクチャを組み付け、それによって、ＤＲＡＭメモリの最大バーストレートでのデータ転送を可能にし、実時間ボリュームレンダリングを可能にする。各ＤＲＡＭモジュール内では、連続してアクセスされるミニブロックが異なるバンクに割り当てられるように、ミニブロックがメモリバンクに割り当てられ、それによってデータ転送中のアイドルサイクルを回避し、ＤＲＡＭ転送効率をほぼ１００％に増加させる。



【特許請求の範囲】

【請求項１】　ボクセルデータをＤＲＡＭメモリモジュールに配置して、ボリュームレンダリングシステムにおけるデータ転送中のアイドルサイクルを回避し、それによってＤＲＡＭメモリモジュールの最大バーストレートでのデータ転送による実時間ボリュームレンダリングを可能にする方法であって、

ボクセルデータ集合からのボクセルデータをミニブロックに配置するステップと、

それぞれ多数のＤＲＡＭメモリバンクを有する多数のＤＲＡＭモジュールを提供するステップと、

各ＤＲＡＭモジュール内で、連続的にアクセスされるミニブロックが関連するＤＲＡＭモジュール内で異なったバンクに割り当てられるように、ミニブロックをそのメモリバンクに割り当てるステップであり、上記ミニブロックは、特定のＤＲＡＭと２層記憶システムにおけるＤＲＡＭ内の特定のバンクとの両方に割り当てられ、上記バンクの連続的アクセスがあった時点で、関連するＤＲＡＭモジュールからのデータ転送の間のアイドルサイクルが回避されるステップとを含む実時間ボリュームレンダリングを可能にする方法。

【請求項２】　所定の順序でＤＲＡＭメモリのバンクからボクセルデータを読み出すステップと、上記読み出しがコンフリクトにつながる場合を検出するステップと、上記コンフリクトの検出に応じて読み出し順序を反転させるステップとをさらに含むことを特徴とする請求項１に記載の実時間ボリュームレンダリングを可能にする方法。

【請求項３】　上記コンフリクトは、現在読み出されているのと同一メモリバンクからの連続的読み出しを含むことを特徴とする請求項２に記載の実時間ボリュームレンダリングを可能にする方法。

【請求項４】　ＤＲＡＭメモリから読み出されたボクセルデータを再度順序付けして、上記ボクセルが上記ボリュームデータ集合に配置された順序でのボクセル処理を可能にするステップをさらに含むことを特徴とする請求項１に記載の実時間ボリュームレンダリングを可能にする方法。

【請求項５】　上記ミニブロックのそれぞれは、２×２×２ボクセルの立体配列での８ボクセルを含むことを特徴とする請求項１に記載の実時間ボリュームレンダリングを可能にする方法。

【請求項６】　上記ミニブロックは、次式に従って上記ＤＲＡＭモジュールに記憶され、

ＤＲＡＭ Module Number＝（$X_{mb}+Y_{mb}+Z_{mb}$）mod Ｍ

ここで、$X_{mb}+Y_{mb}+Z_{mb}$は次式によって求められる座標（Ｘ、Ｙ、Ｚ）を有するボクセルを含むミニブロックの位置を定義する。

X_{mb}＝［Ｘ／２］，Y_{mb}＝［Ｙ／２］，Z_{mb}＝［Ｚ／２］

Ｍ個のモジュールの全ては、１つのモジュールにアクセスするのに必要な時間と同じ長さで同時にアクセスでき、ミニブロックの座標を合計し、Ｍで割って剰余を取ることによって、上記ボリュームデータ集合の何れかの軸と同一線上のＭ個のブロックの何れかのグループが一斉かつ同時にフェッチされることを保証することを特徴とする請求項１に記載の実時間ボリュームレンダリングを可能にする方法。

【請求項７】　座標（X_{mb}，Y_{mb}，Z_{mb}）を有するミニブロックは、次式に従ってバンクに割り当てられる

Bank Number＝｛［X_{mb}／Ｍ］＋［Y_{mb}／Ｍ］＋Z_{mb}／Ｍ］｝mod Ｂ

ここで、Ｍは独立のメモリチップの数であり、Ｂは１チップ当たりのバンクの数であり、また、ＢはＤＲＡＭチップ１個当たりのバンクの数である。ことを特徴とする請求項６に記載の実時間ボリュームレンダリングを可能にする方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】この発明は、実時間ボリュームレンダリングを可能にする方法に関し、ボリュームレンダリング、より詳細には、ボリュームデータ集合の記憶に最小サイズのデータブロックを用いてメモリの迅速な読み出しを介して実時間ボリュームレンダリングを可能にするメモリアーキテクチャに関する。

【０００２】

【従来の技術】ボリュームレンダリングは、３以上の次元におけるサンプルデータとして表すオブジェクトまたは事象の視覚化を扱うコンピュータグラフィックスの分野である、ボリュームグラフィックスの一部である。これらのサンプルはボリュームエレメント、すなわち「ボクセル」と呼ばれ、検討されるオブジェクトや事象の物理的特徴を表すデジタル情報を含む。ボリュームレンダリングは、印刷、コンピュータ端末での表示および他の形態の視覚化を目的として、二次元画像としてのボリュームデータの投影に関係するボリュームグラフィックスの一部である。実時間ボリュームレンダリングは、通常は１秒間に３０フレーム以上の、高速で連続する一連の画像としてのボリュームデータの投影および表示であり、これによって、オペレータがユーザーに視覚的なフィードバックを逐一提供しながら投影のパラメータを対話的に制御し、画像を操作することを可能にする。

【０００３】ボリュームレンダリング用のソフトウェア手法は１０年から２０年に亘って実施されてきたが、要求される膨大な計算能力と、十分な速度でボクセルデータを読み出し・移動することの困難さとの両方の理由から、実時間ボリュームレンダリングには適していなかった。現代のパーソナルコンピュータの急速な能力向上をもってしても、今後何年にも亘ってソフトウェア上で実時間ボリュームレンダリングを支援するには十分とはな

りそうもない。例えば、各エッジに２５６ボクセルで、合計で256^3または１６００万ボクセルあるボリュームデータ集合を表現し、これを実時間で行うには、１秒当たり３０回以上で全ての１６００万ボクセルを読み出し処理することが必要である。これは１秒間に５億ボクセルを超えるレートで読み出しおよび処理をしなければならないことになり、このレートは現代のパーソナルコンピュータで利用できる計算能力およびメモリ帯域幅を遥かに超えるものである。512^3ボクセルのボリュームデータ集合は、その８倍の（読み出しおよび処理）レート、すなわち１秒間に約４０億ボクセルのレートを必要とし、1024^3ボクセルのボリュームデータ集合はさらにその８倍の（読み出しおよび処理）レート、すなわち１秒間に約３２０億ボクセルのレートを必要とすることがわかる。各フレームにおいて処理されるボクセルの数を削減するために確立されたソフトウェア技術を使用したとしても、レートは依然として現代のパーソナルコンピュータのメモリ帯域幅および計算能力を超える。

【０００４】しかし、現代の半導体技術を用いれば、特殊なの目的のためのボリュームレンダリングシステムを、例えば回線基盤にプラグインで取り付けるパーソナルコンピュータの付属品として構築することが可能になった。かかるシステムにおいては、ボクセルはＤＲＡＭチップとも呼ばれる複数のダイナミックランダムアクセスメモリーのモジュールに記憶される。データは１つ以上の平行なパイプライン化された処理エレメントによって読み取りおよび処理され、実時間のフレームレートで画像を投影する。かかる特殊なシステムでは、いかにして十分な速度でメモリからボクセルデータを読み出すかが難問となる。この速度は、バーストモード、すなわち隣接したメモリアドレスに記憶された一連のデータ値を連続的にすばやく読み出すモード、で動作する最速のＤＲＡＭチップを除けば、すべてのメモリの帯域幅を超えている。バーストモードのＤＲＡＭチップであったとしても、メモリの効率をほぼ１００％まで最大化し、定格帯域幅のほぼ１００％でチップを作動させることが必要となる。

【０００５】１９９７年８月１日に出願されており、本明細書中に参考文献として含めた米国特許出願第０８／９０５，２３８号において、実時間ボリュームレンダリングシステムは、一つのブロック内のすべてのボクセルが単一のメモリモジュール内の隣接したメモリアドレスに逐次格納されるように、ボクセルデータがブロックに編成されている、と説明されている。これによって、一度に一つというよりも、ブロック全体のデータを一気にフェッチすることが可能となり、それによってＤＲＡＭに関連するバーストモードのアクセスを利用することができる。ボクセルのブロックがフェッチされると、ボクセルはパイプライン毎に１サイクル当たり１ボクセルのレートで、１つ以上の処理パイプラインに送られる。その間に、それに続くボクセルのブロックのフェッチングが始まる。典型的な高性能ＤＲＡＭチップは、それぞれ７．５ナノ秒、７ナノ秒および６ナノ秒のサイクル時間に対応して、１秒間に１億３３００万、１億４７００万または１億６６００万のレートで動作させることができる。各ボリュームの値がそれぞれ１つのＤＲＡＭデータエレメントからなるとすると、１秒間に５億ボクセルという必要データレートを達成するために、およそ４つのＤＲＡＭチップを並行して動作させる必要がある。

【０００６】ボクセルデータのブロックを読み出す順序は、ボリュームデータ集合を見る方向、すなわちボリュームデータ集合に対する画像平面の位置によることがお分かり戴けるであろう。何れの見る方向についても必要なボクセル読み出しおよび処理レートを達成するため、ＤＲＡＭチップの並行操作にコンフリクトがないように、実時間ボリュームレンダリングシステムのＤＲＡＭチップ全体に渡ってボクセルデータを分散することが必要である。これは、Ｃｕｂｅ－４というシステムで実施されており、１９９６年１２月にHanspeter PfisterがStony Brookのニューヨーク州立大学のコンピュータサイエンス学部に提出した「Architectures for real-time Volume Rendering」という題の博士論文に記載され、さらに米国特許第５，５９４，８４２号「実時間ボリューム視覚化のための装置と方法」に記載されたボクセルデータを「スキューイング：skewing」する方法によって達成される。このスキューイングの方法は改良され、米国特許出願第０８／９０５，２３８号に記載された、ＥＭ－Ｃｕｂｅというシステムにおけるボクセルのブロックのメモリ組織に適合された。

【０００７】Ｃｕｂｅ－４システムのスキューイングの本質は、隣接するボクセルが異なったＤＲＡＭチップに記憶されているということである。これは３つの全ての次元にあてはまり、従って同じ数のＤＲＡＭチップから、ボリュームデータ集合の何れかの軸と一線上に並べられた隣接するボクセルの何れかのグループを同時にフェッチすることが可能である。これによってＤＲＡＭチップを並行に使用する効率は最大化されるが、各ＤＲＡＭチップの帯域幅を非効率的に利用することになる。ＥＭ－Ｃｕｂｅシステムの本質は、個々のボクセルではなくて隣接するボクセルのブロックが、隣接するＤＲＡＭチップに記憶されることである。これによって各ＤＲＡＭチップの帯域幅の効率は改善するが、ＤＲＡＭチップがバンクに編成される方式のために、改善の程度はブロックのサイズに左右される。

【０００８】特に、現代のＤＲＡＭチップは複数のバンクのメモリからなり、各バンクは複数の列からなり、各列は連続的なメモリアドレスにおける複数のデータエレメントからなっている。かかるＤＲＡＭチップは、単一のバンクの単一の列内においてデータを読み書きしている間に、その最大レートの帯域幅を持続することができ

る。同時に、異なったバンクの列は「プリチャージする：pre-charged」か転送の準備をすることができ、従って、読み書きは直前に使用したバンクの直前の列から新たに使用するバンクの新たな列へと中断なく継続して行うことができる。しかし、ＤＲＡＭチップは同一バンクの２つの異なる列を素早く連続して読み取ったりそこに書き込んだりすることはサポートしていない。すなわち、同一バンクの異なる列から読み出したりそこに書き込んだりしている間に、バンクの１つの列をプリチャージすることは不可能である。ＤＲＡＭには付加的な制限を課すもの、例えば、使用中のバンクに隣接したバンクのプリチャージを禁ずるものもある。

【０００９】同一の、もしくは対立するバンクの異なる列を読み出したりそこに書き込んだりしなければならないような形でデータが編成されている場合には、常に、数サイクルの遅延が課される。実時間ボリュームレンダリングシステムにおいては、この遅延による影響はブロックのサイズに応じて変わる。例えば、ＥＭ－Ｃｕｂｅシステムの実施の形態においては、ブロックは８×８×８ボクセル、すなわち合計で５１２ボクセルである。この場合に、同一バンクの列の間で８サイクルの遅延があるＤＲＡＭを使用すると、約９７％の効率でＤＲＡＭチップからボクセルデータを読み取ることが依然として可能である。しかし、それよりも少ない２×２×２のボクセル、すなわち合計で８ボクセルを有する別の実施の形態においては、ＤＲＡＭ帯域幅の効率は約５０％に削減されるであろう。ここで、実時間ボリュームレンダリングシステムに関する試みは、ブロックを十分に大きく保つか、同一のもしくは対立するバンクの異なった列への迅速かつ連続的なアクセスを避けるかの何れかによって、ＤＲＡＭメモリの効率を最大化するようにデータを編成することである。

【００１０】

【発明が解決しようとする課題】上記従来のシステムは、通信効率を最大化するために比較的大きなボクセルデータのブロックを利用しているが、現在単一の集積回路またはチップ上でボリュームレンダリングシステムを実施することが望ましくなっている。しかし、実時間ボリュームレンダリングのパフォーマンスを達成するためには、ボクセルの記憶および分散という根本的なアーキテクチャにおける変更が必要である。

【００１１】この発明は上述した点に鑑みてなされたもので、実時間ボリュームレンダリングのパフォーマンスを達成できる実時間ボリュームレンダリングを可能にする方法を提供することを目的とするものである。

【００１２】

【課題を解決するための手段】この発明に係る実時間ボリュームレンダリングを可能にする方法は、ボクセルデータをＤＲＡＭメモリモジュールに配置して、ボリュームレンダリングシステムにおけるデータ転送中のアイドルサイクルを回避し、それによってＤＲＡＭメモリモジュールの最大バーストレートでのデータ転送による実時間ボリュームレンダリングを可能にする方法であって、ボクセルデータ集合からのボクセルデータをミニブロックに配置するステップと、それぞれ多数のＤＲＡＭメモリバンクを有する多数のＤＲＡＭモジュールを提供するステップと、各ＤＲＡＭモジュール内で、連続的にアクセスされるミニブロックが関連するＤＲＡＭモジュール内で異なったバンクに割り当てられるように、ミニブロックをそのメモリバンクに割り当てるステップであり、上記ミニブロックは、特定のＤＲＡＭと２層記憶システムにおけるＤＲＡＭ内の特定のバンクとの両方に割り当てられ、上記バンクの連続的アクセスがあった時点で、関連するＤＲＡＭモジュールからのデータ転送の間のアイドルサイクルが回避されるステップとを含むものである。

【００１３】また、所定の順序でＤＲＡＭメモリのバンクからボクセルデータを読み出すステップと、上記読み出しがコンフリクトにつながる場合を検出するステップと、上記コンフリクトの検出に応じて読み出し順序を反転させるステップとをさらに含むことを特徴とするものである。

【００１４】また、上記コンフリクトは、現在読み出されているのと同一メモリバンクからの連続的読み出しを含むことを特徴とするものである。

【００１５】また、ＤＲＡＭメモリから読み出されたボクセルデータを再度順序付けして、上記ボクセルが上記ボリュームデータ集合に配置された順序でのボクセル処理を可能にするステップをさらに含むことを特徴とするものである。

【００１６】また、上記ミニブロックのそれぞれは、２×２×２ボクセルの立体配列での８ボクセルを含むことを特徴とするものである。

【００１７】また、上記ミニブロックは、次式に従って上記ＤＲＡＭモジュールに記憶され、

ＤＲＡＭ Module Number＝（X_{mb}＋Y_{mb}＋Z_{mb}）mod Ｍ

ここで、X_{mb}＋Y_{mb}＋Z_{mb}は次式によって求められる座標（Ｘ、Ｙ、Ｚ）を有するボクセルを含むミニブロックの位置を定義する。

X_{mb}＝［Ｘ／２］，Y_{mb}＝［Ｙ／２］，Z_{mb}＝［Ｚ／２］

Ｍ個のモジュールの全ては、１つのモジュールにアクセスするのに必要な時間と同じ長さで同時にアクセスでき、ミニブロックの座標を合計し、Ｍで割って剰余を取ることによって、上記ボリュームデータ集合の何れかの軸と同一線上のＭ個のブロックの何れかのグループが一斉かつ同時にフェッチされることを保証することを特徴とするものである。

【００１８】また、座標（X_{mb}，Y_{mb}，Z_{mb}）を有するミニブロックは、次式に従ってバンクに割り当てられる

Bank Number＝｛［X_{mb}／M］＋［Y_{mb}／M］＋Z_{mb}／M］｝mod B

ここで、Mは独立のメモリチップの数であり、Bは１チップ当たりのバンクの数であり、また、BはＤＲＡＭチップ１個当たりのバンクの数である。ことを特徴とするものである。

【００１９】

【発明の実施の形態】以下、この発明について説明する。この発明に係るシステムにおいては、ＤＲＡＭメモリモジュールとの間のデータ転送がほぼ１００％のレベルにまで確実に最大化されるように、ボクセルデータのミニブロック編成が採用されている。ミニブロック編成は、各ミニブロックがＤＲＡＭメモリモジュールまたはチップに割り当てられ、さらにＤＲＡＭメモリモジュール内の特定のバンクに割り当てられるようになっている。これは、メモリモジュールのみならずそこに含まれたバンク全体に渡ってボクセルデータの２層の割り当てすなわちスキューイングを構成する。メモリアーキテクチャは、削減されなければ、ＤＲＡＭモジュールの定格帯域幅の５０％以下にまで大幅にデータレートを減少させてしまう、アイドルサイクルの数を最少値にまで削減する。

【００２０】ボクセルデータはミニブロックに記憶されているので、ＤＲＡＭモジュールの最大データレートバーストで読み出すことができる。ミニブロックはＤＲＡＭのバンク全体に渡って分散されるかスキューされるので、メモリバンクをプリチャージするのに必要なアイドルサイクルを生じさせることなく、連続したミニブロックを読み出すことが可能になる。周知のように各アイドルサイクルは、それがなければデータを読むことができる（実際には読まれない）損失時間の単位を表す。このアイドルサイクルは、メモリから読み出すデータの効率を削減し、実時間ボリュームレンダリングシステムの費用を増加させる、もしくは全く不可能にしてしまう。従って、このアイドルサイクルを回避することが重要である。

【００２１】各ＤＲＡＭチップのバンク全体に渡るミニブロックのスキューイングをもってしても、１つの列の終端で１つのＤＲＡＭの１つのバンクからミニブロックをフェッチした直後に、次の列の始点で同一ＤＲＡＭモジュールの同一バンクからミニブロックを連続してフェッチしてしまうといった事態がしばしば発生する。この問題を回避するために、この発明は、読み出される次のミニブロックがＤＲＡＭメモリの同一または対立するバンクからのフェッチにつながる場合には、ミニブロックの列の読み出し方向を反転するシステムを利用している。

【００２２】ボクセルデータは割り当ての順序でＤＲＡＭメモリモジュール全体に渡って分散されるのであって、特定の観察方向に関する通常の処理ではないので、ＤＲＡＭモジュールの出力を再配置して所与の観察方向に関する通常の処理順序を再確立するために、特化されたデスキューイング回路が提供される。これは、一つの実施の形態においては多くの内部バッファおよびアドレッシングシステムの制御下でのシフティングまたはマルチプレクシング回路によって達成される。

【００２３】より詳細には、この発明においては、実時間ボリュームレンダリングに必要とされるパイプライン化された処理エレメントは、単一の半導体チップに設けられた単一の集積回路に含まれている。他の処理チップとの通信はなく、従って、かかる通信の帯域幅を管理する必要もないので、大規模なブロックの必要性は抹消される。従って、この発明は、ＤＲＡＭチップのバーストモード転送を利用することができる最小のブロックを使っている。より好適な実施の形態においては、ブロックは２×２×２であり、従ってミニブロックと呼ばれている。ミニブロックは、前述のＥＭ－Ｃｕｂｅボリュームレンダリングシステムにおけるのと全く同じ方法で、メモリのミニブロックの位置を示す同一の数学的公式をもって、ボリュームデータ集合全体に渡ってスキューされる。

【００２４】ＤＲＡＭの同一バンクからの連続的なフェッチの結果生ずるＤＲＡＭ帯域幅の効率低下を避けるために、この発明は、ミニブロックがＤＲＡＭチップ全体に渡ってスキューされ、また各ＤＲＡＭチップ内のバンク全体にも渡ってスキューされる、２層のスキューイング手法を利用している。すなわち、隣接するミニブロックが別個のＤＲＡＭチップに記憶され、各ＤＲＡＭチップ内においては、ボリュームデータ集合の３本の軸に平行な何れかの方向で、至近隣接するバンクから離れたバンクにミニブロックが記憶されている。レンダリングの間には、ミニブロックはグループでフェッチされ、グループ内のミニブロックの数は、システムにおいて同時にアクセス可能なＤＲＡＭチップの数と同じである。ミニブロックのグループは、例えば好適な実施の形態において、ボリュームデータ集合が画像平面から観察される時に左から右、上から下、正面から背面へとステップすることにより、秩序だった方法でフェッチされる。２層のスキューイングによって、何れかのＤＲＡＭチップにおける連続的なミニブロックは異なったバンクからフェッチされ、従って、ＤＲＡＭの帯域幅の効率が最大化されることが確実になる。

【００２５】しかし、ミニブロックの各配置、スキューイングの方法、およびボリュームデータ集合を介した処理の順序に関しては、ミニブロックの列の最後のミニブロックが、ミニブロックの次の列の最初のミニブロックと同じバンクに記憶されるような、少なくとも１つの方向付けまたは観察方向がある。特別の配慮がなければ、このような１つの事態が処理の遅延、ＤＲＡＭ帯域幅の浪費効率、またはパイプライン化された処理エレメント

における付加的な複雑性を発生させることになる。したがって、この発明は、これらの特定の列を反対の順序で処理することにより、このような事態を補正するものである。すなわち、例えば左から右へとミニブロックの列がフェッチされ、次の列が前の列と同一バンクにあるミニブロックで始まる場合には、次の列は右から左にフェッチされる。この手段によって、ミニブロックへの全ての連続的なフェッチは、ＤＲＡＭチップのコンフリクトしない、異なるバンクから行われることになる。

【００２６】ミニブロックがその対応するＤＲＡＭチップからフェッチされるときに、デスキューイング回路、すなわち観察方向に従ってボクセルを再配置し、各ボクセルをそれを処理する処理パイプラインで整列させる回路を通過させられる。幾つかの実施の形態においては、各ボクセルは正確に１つの処理パイプラインと関連づけられており、隣接するボクセル値は、隣接する処理パイプラインからまたは処理チップのＦＩＦＯを介して再循環されるボクセルデータからの通信によって得られる。しかし他の実施の形態においては、ボクセルキャッシュとよばれる、処理チップ上の記憶バッファが利用される。これらの実施の形態においては、パイプラインは互いには全く通信せず、何らかの特定の処理に必要なボクセルをボクセルキャッシュからフェッチする。何れのアプローチにおいても、デスキューイング回路およびボクセルキャッシュが、各処理パイプラインが正しいデータを正しい順序で検索することを可能にしている。

【００２７】要約すると、２層のスキューイングアーキテクチャが、ボクセルデータが１組のＤＲＡＭメモリモジュールに割り当てられたミニブロックに記憶されている、ボリュームレンダリングシステムのメモリサブシステムに課されており、それによって、ＤＲＡＭメモリの最大バーストレートでのデータ転送を可能にし、実時間ボリュームレンダリングを可能にしている。各ＤＲＡＭモジュール内においては、連続的にアクセスされるミニブロックが異なったバンクに割り当てられるように、ミニブロックが複数のメモリバンクに割り当てられており、それによって、データ転送中のアイドルサイクルを回避してＤＲＡＭ転送効率をほぼ１００%に増加させる。一つの実施の形態においては、ＤＲＡＭメモリのバンクからのボクセルデータの読み出しは、バンクのコンフリクトがなければ左から右に進行し、バンクのコンフリクトがあれば読み出し順序は反転する。特化されたデスキューイング回路は、ボクセルがメモリに記憶されている順序ではなく、ボリュームデータ集合に配置された順序でボクセルを処理できるように、ＤＲＡＭメモリから読み出されたボクセルデータを再配置するために提供する。

【００２８】より詳細には、低コストな実時間ボリュームレンダリングシステムの記憶サブシステムは、バーストモードのダイナミックランダムアクセスメモリ（ＤＲＡＭ）集積回路または「チップ」のピークレートに近似の持続するデータレートでボリュームエレメントまたは「ボクセル」を配布する。このデータレートは観察方向、すなわちボリュームデータ集合を通過する光線の方向からは独立しており、システムにおけるＤＲＡＭチップの数に比例して一次元的に増減する。三次元のボクセルデータは２×２×２のサイズのミニブロックに編成される。観察方向からの独立性は、隣接するミニブロックが隣接するＤＲＡＭチップに記憶されるように、ＤＲＡＭモジュール全体に渡ってミニブロックをスキューすることにより達成される。これは、あらゆるクロックサイクルにおいて全てのチップから並行してデータを読み取ることができること、および、データは再度読み取る必要がないことを保証し、それによってメモリ帯域幅を最大化する。各ＤＲＡＭチップ内においては、ミニブロックはそのチップのメモリバンク全体に渡ってスキューされる。これは、同一バンクの異なるページへの連続的なフェッチにつきもののプリチャージおよび再起動時間を削減する。一次元的に増減するデータレートのスケーラビリティは、システムにおけるＤＲＡＭモジュールの数を増加させることにより達成できる。ボリュームレンダリングシステムのメモリサブシステムは、パイプライン毎にＤＲＡＭチップサイクルにつき１ボクセルのレートでレンダリングパイプラインにボクセルを提供するために、ミニブロックをデスキューする。

【００２９】パイプライン化されたボクセルレンダリング

図１を参照しさらに背景を説明すると、三次元ボリュームデータ集合１０が図示されている。図１は矩形の立体の形状で配置されたボクセル位置１２を示す。より詳細には、ボクセル位置は三次元で立体を満たしており、均一に間隔が隔てている。各ボクセルの位置は、Ｘ、ＹおよびＺという３本の軸１１によって定められた座標系で表示することができる。各ボクセル位置には、例えばオブジェクトまたはシステムの内部全体および近傍の空間の別個の点における密度、材料の種類、温度、速度、不透明度や他の属性などの、検討中のオブジェクト、システムまたは事象の幾つかの特徴を表す１つ以上のデータ値が関連づけられている。コンピュータ内部では、配列インデックス位置（Ｘ、Ｙ、Ｚ）を三次元空間の座標（Ｘ、Ｙ、Ｚ）におけるボリュームデータ値に対応させ、三次元配列の値としてボリュームデータ集合をコンピュータに表示する方法が便利である。

【００３０】図２は、人間の頭部の断層撮影スキャンによるスライスからなるボリュームデータ集合１０の例を示す。二次元画像平面１６は、人間の頭部のボリュームレンダリングされた投影が表示された表面を示している。光線キャスティングとして公知の技術においては、光線１８がボリュームデータ集合１０を通って画像平面１６のピクセル位置２２から投影され、各光線はボリュ

ームを通過するときに、ボクセル位置のデータから色と不透明度を蓄積する。このようにして、色、透明度、輝度並びにピクセルのその他の属性が、光線に沿ったサンプルポイント２０におけるデータの蓄積としてボリュームデータ集合から抽出される。この例においては、骨組織と関連するボクセル値には不透明色が割り当てられ、頭部の他の全ての組織に関連するボクセル値には透明色が割り当てられる。従って、光線に沿ったデータの蓄積および、対応するピクセルへのこのデータの帰属は、実際の頭蓋は皮膚や頭部の他の組織によって視界からは隠れていても、観察者には三次元の頭蓋の画像に見える画像１９を観察面１６に形成することになる。

【００３１】光線キャスティング（ray-casting）の方法をより完全に理解するために、図３は図２の三次元ボリュームデータ集合１０の二次元断面を示している。第一および第二の次元はこのページの平面に図示された次元に対応する。ボリュームデータ集合１０の第三の次元は、データ集合の断面のみを図において見ることができるように、印刷されたページに垂直である。ボクセル位置は図の点１２によって示されている。各位置に関連したボクセルは、三次元空間における矩形グリッドの固定点での三次元オブジェクト１４のある特徴を表すデータ値である。図３には、適当な特徴を備えたピクセル２２を提供するという意味で、オブジェクト１４の画像が投影される二次元画像平面１６の一次元の図を示す。この図においては、画像平面１６の第二の次元も印刷されたページに垂直である。

【００３２】光線キャスティングの技術において、光線１８は画像平面１６のピクセル２２からボリュームデータ集合１０を通って伸びている。各光線は、その光線に沿ってサンプルポイント２０において色、輝度および透明度または不透明度を蓄積する。この光の蓄積が、対応するピクセル２２の明るさおよび色を決定する。従って、ボリュームを通ってピクセルから外方に進むと図示されている一方で、蓄積されたデータは光線を戻って送信され、対応するピクセルにパラメータ、とりわけピクセルの色、輝度および不透明度または透明度を与えられるように対応するピクセルに提供されると考えることができる。

【００３３】図３はボリュームデータ集合１０の第三の次元と画像平面１６の第二の次元が、共に印刷ページに垂直であり、従って互いに平行であると示唆しているが、一般的にはそうではないということである。画像平面はボリュームデータ集合に対していかなる方位も有することができ、従って光線１８は三次元すべてにおいてどんな角度でもボリュームデータ集合１０を通過する。

【００３４】また、サンプルポイント２０はボクセル１２の座標と必ずしも正確に交差するとは限らないことがわかる。従って、各サンプルポイントの値は、隣接するボクセルの値から合成しなければならない。すなわち、各サンプルポイント２０における光の輝度、色および透明度または不透明度を、隣接するボクセル１２の値の数学的関数によって計算するか、もしくは補間する必要がある。ボクセルデータ値のサンプルポイントにおける値への再サンプリングは、サンプリング理論として知られる数学部門の応用である。そして、各光線１８のサンプルポイント２０は、もうひとつ数学的関数によって蓄積され、その光線に対応するピクセル２２の輝度および色が求められる。その結果得られるピクセル２２の集合は、オブジェクト１４の視覚画像を画像平面１６に形成する。

【００３５】図４は個々の光線の処理を示している。光線１８はボクセル位置１２の付近またはその位置を通って、三次元ボリュームデータ集合１０をある角度で通過し、各光線に沿ったサンプルポイント２０でデータを蓄積する。２１に示すように、各サンプルポイントにおける値は、補間（内挿）ユニット１０３（図５参照）によって合成され、その勾配は２３に示すように勾配推測ユニット１１１（図５参照）によって計算される。サンプルポイント２０からのサンプルポイント値と各サンプルポイントに関する勾配２５は次に、パイプラインにおいて処理されて各サンプルに色、輝度または強さ、および透明度または不透明度を割り当てる。２７に示すように、これはパイプライン処理を経て、赤、緑および青の色相ならびに輝度および透明度または不透明度が計算される。最後に、全ての光線のすべてに沿ったサンプルのすべてに割り当てられる色、輝度レベルおよび透明度が、２９に示すように合成ユニット１２４に適用される。合成ユニット１２４は、これらのサンプル値を数学的に結合してピクセルと成し、ピクセルは結果として生ずる画像３２を画像平面１６上に描く。

【００３６】サンプルポイント２０の色、輝度または強さおよび透明度の計算は、２つに分けて行われる。第一に、三線補間の数学的関数を利用して、サンプルポイント２０を隣接して囲むキュービック配置内の８つのボクセルの値の加重平均を取る。その結果得られる平均はを使って、ある伝達関数によって色および不透明度または透明度をサンプルポイントに割り当てる。第二に、隣接のサンプルポイント間の差を取ることによって各サンプルポイント２０におけるサンプル値の数学的勾配を概算する。理解されるように、これら２つの計算は数学的に均等な結果を得るために、順番にもしくは互いに並行して実行することが可能である。勾配は次に、サンプルポイントの明るさを判断するための照明計算に使用される。照明計算はコンピュータグラフィックスの分野においては周知であり、例えば、Addison-Wesley of Reading, Massachusettsによって１９９０年に発行された、J. Foley, A. vanDam, S. FeinerおよびJ. Hughes著の教科書「Computer Graphics:Principles and Practice」第２版で説明されている。

【００３７】図５は、図４に示した計算を行うのに適したパイプライン化されたプロセッサのブロック図を示している。パイプライン化されたプロセッサは複数のパイプライン段階からなり、各段階は１つのデータエレメントを保持しており、従って、複数のデータエレメントが一度に処理される。各データエレメントはその処理においては進捗度合いが異なっており、全てのデータエレメントがパイプラインの段階から段階へとロック（固定）ステップで移動する。パイプラインの第一の段階では、一連のボクセルデータ値は１サイクル当たり１ボクセルのレートでボクセルメモリ１００からパイプラインに流れ込み、アドレスジェネレータ１０２の制御下で機能する。補間（内挿）ユニット１０４は、三次元空間のＸ、ＹおよびＺ座標にあるボクセル値を受け取る。ここでＸ、ＹおよびＺはそれぞれ整数である。補間（内挿）ユニット１０４は、パイプライン化された一組の段階であり、ボリュームを通って放たれる光線に沿った位置に対応するボクセル間のデータ値をサンプルポイントで合成する。各サイクルの間に、１つのボクセルが補間（内挿）ユニットに入ると、１つの補間（内挿）されたサンプル値が現れる。ボクセルがパイプラインに入ってから補間（内挿）されたサンプル値が現れるまでの待ち時間は、パイプライン段階の数と各段階での内部遅延に左右される。

【００３８】パイプラインの補間（内挿）段階は、一組の補間（内挿）段階１０４と、段階を経てデータを再循環するための３個のＦＩＦＯエレメント１０６、１０８、１１０からなる。一つの実施の形態においては、これらは全て線形補間（内挿）であるが、立方（cubic）やラグランジアン（LaGrangian）などの他の補間（内挿）関数を採用してもよい。図示した実施の形態においては、補間（内挿）は各次元において個別の段階として行われ、空間では隣接しているがパイプラインに入る時点では全く離れているボクセル間で補間を行えるよう、データを再循環するための個々のＦＩＦＯエレメントが含まれている。各ＦＩＦＯの遅延は、特定の次元における１つのボクセルの読み取りと隣接するボクセルの読み取りとの間に経過した時間の長さに正確に一致するよう選択され、これら２つのボクセルが補間関数において結合できるようになっている。理解されるのは、各ボクセルがその次元に関連したＦＩＦＯを通ってあらかじめ再循環された最も近接するボクセルと結合下状態で、１サイクル当たり１ボクセルのレートでボクセルを補間段階に通過させることができる。

【００３９】各次元に１つずつの３つの連続的な補間段階は連結されており、ボクセルは入力と出力の両方において、１サイクル当たり１ボクセルのレートでこれら３つの段階を通過することができる。補間段階のスループットは、補間（内挿）ユニット内の段階の数に関係なく、また、補間（内挿）ユニット内のデータ待ち時間およびユニット内の再循環段階の待ち時間に関係なく、１サイクル当たり１ボクセルである。従って、補間（内挿）ユニットは１サイクル当たり１ボクセルのレートで、ＸＹＺ空間の整数位置にあるボクセル値を、非整数位置にあるサンプル値に変換する。より詳細には、補間（内挿）ユニットは、ボクセル位置の値を光線に沿って配置されたサンプル位置の値に変換する。

【００４０】補間（内挿）ユニット１０４に続くのは勾配推測ユニット１１２であり、これも複数のパイプライン化された段階と再循環ＦＩＦＯからなっている。勾配ユニット１１２の機能は、３つの各次元におけるサンプル値の変化レートを推測することである。勾配推測ユニットは補間（内挿）ユニット１０４と同様の方法で機能し、３つの次元のそれぞれにおけるサンプル値の変化レートを計算する。勾配は照明に関する通常のベクトルを判断するのに使用され、その大きさは、勾配の程度が高い場合に表面の存在を測るものさしとして使用することがある点に注意して戴きたい。本実施の形態においては、中央の差を取ることにより得られるが、当該技術分野で公知な他の関数を採用してもよい。

【００４１】勾配推測ユニットはパイプライン化されているので、１サイクル当たり１つの補間されたサンプルを受信し、１サイクル当たり１つの勾配を出力する。補間（内挿）ユニット同様、各勾配は、個々の再循環ＦＩＦＯ１１４、１１６、１１８を含む勾配推測ユニット１１２における待ち時間の長さに等しいサイクルの数によって、その対応するサンプルから遅延る。各再循環ＦＩＦＯの遅延は、その次元において勾配を演繹するのに必要な１つの補間されたサンプルと近接する付近の補間されたサンプルと、読み取りに必要な時間の長さによって決定される。

【００４２】補間されたサンプルとそれに対応する勾配は、１サイクル当たり１つの補間されたサンプルにつき１つの勾配のレートで、それぞれ分類ユニット１２０および照明ユニット１２２に同時に適用される。分類ユニット１２０は、補間されたサンプル値をグラフィックシステムにおける色、すなわち赤、緑、青と、ＲＧＢＡ値としても知られるアルファ値に変換する役割を果たす。赤、緑および青の値は通常、ゼロから１までの間の分数であり、個々の補間されたサンプル値に割り当てられる色部品（カラーコンポーネント）の輝度を表す。アルファ値も通常、ゼロから１までの間の分数であり、個々の補間されたサンプル値に割り当てられる不透明度を表す。

【００４３】勾配は照明ユニット１２２に適用されて、強調と影を付加することにより新たに割り当てられたＲＧＢＡ値を調整して、より現実的な画像を提供する。照明を行う方法と機能は当該技術分野においては公知である。照明ユニットおよび分類ユニットは１サイクル当たり１つの補間されたサンプル値と１つの勾配を受け入

れ、１サイクル当たり１つの照明された色（illuminated color）と不透明度の値を出力する。

【００４４】当該の実施の形態においては、補間（内挿）ユニット１０４は勾配推測ユニット１１２より先行し、勾配推測ユニット１１２は分類ユニット１２０より先行する、他の実施の形態においてはこれら３つのユニットは異なる順序で配置してもよいことが理解される。より詳細には、ボリュームレンダリングの幾つかの適用において、分類ユニットは補間（内挿）ユニットに先行することが望ましい。この場合に、ボクセル位置のデータ値は同じ位置でＲＧＢＡ値に変換され、次にこれらのＲＧＢＡ値は補間されて光線に沿ったサンプルポイントでＲＧＢＡ値を得る。

【００４５】合成ユニット１２４は光線に沿って全てのサンプルポイントの照明された色（illuminated color）および不透明度の値を結合し、コンピュータ端末または二次元画像表面に表示するために、その光線に対応する最終ピクセル値を形成する。ＲＧＢＡ値は１サイクル当たり１ＲＧＢＡ値のレートで合成ユニット１２４に入り、同じ光線に沿った以前のサンプルポイントにおけるＲＧＢＡ値とともに蓄積される。蓄積が完了すると、最終の蓄積値がピクセルとしてディスプレイに出力されるか画像データとして記憶されるかする。合成ユニット１２４は１サイクル当たり１ＲＧＢＡのサンプルを受けとり、光線の終端に達するまで合成機能に従って光線ごとこれらの光線を蓄積し、光線の終端に達したところで光線当たり１ピクセルを出力して最終画像を形成する。適用の形態によっては、合成ユニットには当該技術分野で公知の多くの異なる機能を採用することができる。

【００４６】照明ユニット１２２と合成ユニット１２４との間には、多様な調整ユニット１２６を設けて照明されたＲＧＢＡ値の調整を可能にし、それによって最終的に見る画像を調整することができる。かかる調整ユニットの１つは、サンプル値を切り落としてデータの限定されたサブセットを見ることを可能にするのに使用される。他の調整ユニットは、任意の角度および厚さでボリュームデータのスライス部分を表示す機能を提供する。３番目の調整ユニットは、三次元カーソルを提供して、ユーザーまたはオペレータがデータ内のＸＹＺ空間で位置を特定することを可能にする。上記の機能のひとつひとつが、１つのＲＧＢＡ値を１サイクル当たりの入力として受入れ、１サイクル当たり１つの調整されたＲＧＢＡ値を出力として放出する複数のパイプライン化された段階として実施される。本明細書中に記載のパイプライン化されたアーキテクチャ内では、同様に実施することができる他の調整機能も提供され得る。パイプライン化された調整機能を付加することは、処理パイプラインのスループット（レート）を減少させるようなことはいっさいなく、データがパイプラインを通過する際のその待ち時間に影響する。

【００４７】例えば、２５６×２５６×２５６ボクセルを有するボリュームデータ集合についていえば、１秒当たり３０フレームの実時間ボリュームレンダリングレートを達成するためには、ボクセルデータは、１秒当たり$256^3 \times 30$フレームまたは１秒当たり約５億ボクセルでパイプラインに入らなければならない。分かることは、特定のボクセルのいずれに関係する計算も多くの段階を含み、従って定められた待ち時間を有するが、各ボクセルが異なる進捗段階にあり、パイプラインの異なる段階を占めている状態で、複数の異なるボクセルに関連した計算を直ちに行うことが可能である。これによって、計算の複雑さにも拘わらず高い処理レートを維持することが可能になる。

【００４８】さらに、上記のパイプライン化されたプロセッサを複数の平行なパイプラインとして複製して、隣接するボクセルを並行に処理することによりさらに高いスループットレートを達成できることが理解されるであろう。各パイプラインのサイクル時間は、通常のボリュームデータ集合内のボクセル数に求められるフレームレートを掛けパイプライン数で割れば求められる。好適な実施の形態においては、サイクル時間が７．５ナノ秒であり、４つのパイプラインが並行に採用され、それによって１秒当たり５億ボクセル以上もの処理レートを達成する。

【００４９】ミニブロック

図６および図７を参照すると、この発明は上記の米国特許出願第０８／９０５，２３８号に記載のＥＭ－Ｃｕｂｅのブロッキングの方法を利用しており、この方法では、一つの実施の形態において、各ブロックは「ミニブロック」とも呼ばれる２×２×２のサイズの立体配列からなる。図６は軸２０６の座標系に従って三次元空間に配置された８個の隣り合うボクセル２０２の配列２００を示す。８個のボクセル２０２のデータ値は、メモリ２０８の８エレメント配列に記憶される。各ボクセルは、座標（Ｘ、Ｙ、Ｚ）によって表された三次元空間における位置を占め、Ｘ、ＹおよびＺは全て整数である。ミニブロックのメモリ配列内のボクセルデータ値のインデックスは、Ｘ、ＹおよびＺ座標それぞれの順序が下位のビットから判断される。図６に示すように、これら３つの下位の順序のビットは連結されて、ゼロから７までの値となる３ビット二進数２０４を形成し、これは次にそのボクセルに対応する配列エレメントを識別するのに利用される。換言すれば、座標（Ｘ、Ｙ、Ｚ）でのボクセルのデータ値のミニブロック内における配列インデックスは、

$$(X \bmod 2) + 2 \times (Y \bmod 2) + 4 \times (Z \bmod 2) \qquad (1)$$

によって求められる。

【００５０】各ボクセルまたはサンプルの位置が座標

（Ｘ、Ｙ、Ｚ）によって三次元空間で表すことができるように、ミニブロックの位置はミニブロック座標（X_{mb}、Y_{mb}、Z_{mb}）で表すことができる。これらの座標においては、X_{mb}はＸ軸に沿ったミニブロックの位置を表し、全ミニブロックのユニットで数える。同様に、

$$X_{mb}=[X/2],\ Y_{mb}=[Y/2],\ Z_{mb}=[Z/2] \qquad (2)$$

によって求められる。

【００５１】図７を参照すると、ミニブロックスキューイングの最初の層が図示されている。これは次式に従ったＥＭ－Ｃｕｂｅからのブロックのスキューイング方法の応用である。

$$DRAM\ Number=(X_{mb}+Y_{mb}+Z_{mb})\bmod 4 \qquad (3)$$

図においては、ミニブロック２００の三次元配列の部分図が示されており、各ミニブロックは番号を付けた小さな立体によって図示されている。番号はそのミニブロックの特定のＤＲＡＭモジュールチップへの割り当てを表す。図示した実施の形態においては、０、１、２および３と番号を付された４個の異なるＤＲＡＭチップがある。軸と整列した４つのミニブロックの各グループは、４つの番号のそれぞれが付いた１つのミニブロックを含むことが、図から理解される。これは式（３）から確認することができる。すなわち、座標（X_{mb}、Y_{mb}、Z_{mb}）の何れかのミニブロックで始めて、Ｘ軸方向にミニブロックを通って連続すると、式（３）のＤＲＡＭ Ｎｕｍｂｅｒは数字０、１、２および３を経て継続的にサイクルする。同様に、Ｙ軸またはＺ軸に平行なミニブロックを経て連続することにより、式（３）もＤＲＡＭ Ｎｕｍｂｅｒ０、１、２および３を経て継続的にサイクルする。従って、３本の軸の何れかに平行な何れかの方向２０９、２１１または２１３でミニブロックの三次元配列を横断する場合には、４つの隣接するミニブロックのグループは常に、ＤＲＡＭチップの４つの独立したメモリから並行にフェッチできることが理解される。ミニブロックのメモリモジュール内のメモリロケーションへの割り当てを以下に説明する。

【００５２】より一般的には、システムがＭ個の独立したメモリモジュールを含む場合は、座標（X_{mb}、Y_{mb}、Z_{mb}）を有するミニブロックは次のようにメモリモジュールに割り当てられる。

$$Module\ Number=(X_{mb}+Y_{mb}+Z_{mb})\bmod M \qquad (4)$$

すなわち、Ｍ個全てがモジュールにアクセスするのに必要な時間と同じ時間で同時にアクセスすることができるように、図示した実施の形態のメモリサブシステムが、Ｍ個の別モジュールからなっているのであれば、ミニブロックのメモリモジュールへの割り当ては、ミニブロックの座標を合計してＭで割り、剰余を取ることによって求められる。これは、何れかの軸と一線に並んだＭ個のブロックの何れかのグループを、同時にフェッチすることができることを保証する。ボリュームデータ集合の何れかの軸に沿ってＭ個のミニブロックのグループを同時にフェッチする必要があるのは、ボリュームデータ集合の横断の順序が観察方向に依存するためである。

【００５３】図示した実施の形態においては、ミニブロックはボリュームデータ集合の軸と一線上に並んだ線形グループにおいてアクセスされるが、ボリュームデータ集合の横断の順序とは関係なく、矩形グループ、立体グループまたは他の大きさおよび形状のグループにおいてもフェッチできるように、他の実施の形態では異なる公式によってミニブロックをスキューできることが理解されるであろう。

【００５４】現代のＤＲＡＭチップでは、そのＤＲＡＭのタイプに応じたクロックレートで、かつ適度な大きさのバーストで、ＤＲＡＭチップからデータをフェッチしたりＤＲＡＭチップにデータを書き込んだりすることが可能である。いわゆるシンクロナスＤＲＡＭまたは「ＳＤＲＡＭ」と呼ばれるチップには、１３３ＭＨｚ、１４７ＭＨｚおよび１６６ＭＨｚの各種類が含まれており、通常のクロックレートは、１サイクル当たり７．５ナノ秒、７ナノ秒および６ナノ秒にそれぞれ対応する。クロックレートを維持するのに必要な通常のバーストサイズは、一個あたり１６ビットの５個から８個のメモリエレメントである。開発中の別の種類のＤＲＡＭは８００ＭＨｚでのクロックレートと、各１６ビット、通常１６個のデータエレメントのバーストサイズを有する。これら現代のＤＲＡＭチップにおいては、ＤＲＡＭチップ内の独立したメモリバンクからであれば、連続的なバーストはアイドルサイクルの介入なしに適合できる。すなわち、連続してアドレスされたデータエレメントのグループは、ＤＲＡＭチップの異なる、もしくは対立（コンフリクト）しないメモリバンクに記憶され、次にこれらはＤＲＡＭの最大定格速度で、アイドルサイクルの介入も一切なしに、素早く連続して読み取り、書き込みを行える。

【００５５】図８を参照すると、ミニブロックはＤＲＡＭのバンクに対応するグループにさらに配置されている。これはボクセルスキューイングの第二の層を構成する。４ｘ４ｘ４のミニブロックの各グループは大きな数字を付されている。各数字は、そのグループの各ミニブロックの、その割り当てられたＤＲＡＭチップにおける同じ数字を有するバンクへの割り当てを示している。例

えば、図のミニブロック２１２のグループは数字０が付されている。これは、グループ２１２内の各ミニブロックは、メモリチップごとにあるバンク０に記憶されることを意味する。同様に、グループ２１４のミニブロックの全てが、メモリチップごとにあるバンク１に記憶され、グループ２１６の全てがメモリチップごとにあるバンク２に記憶される。

【００５６】図示した実施の形態においては、各ＤＲＡＭモジュールは４つのバンク、すなわち０、１、２および３を付されたバンクを有している。座標（X_{mb}、Y_{mb}、Z_{mb}）を有するミニブロックは、以下の公式に従ってバンクに割り当てられる。

$$Bank\ Number = [\,[X_{mb}/4] + [Y_{mb}/4] + Z_{mb}/4]\,] \bmod 4 \quad (5)$$

図示した実施の形態のにおけるＤＲＡＭチップ毎のバンクの数と、ＤＲＡＭチップの数とが同数であるという事実は単なる偶然である。

【００５７】図から、所与の直交する方向において、一組のパイプライン化された処理エレメントがボリュームデータ集合を横断する場合には、４つのミニブロックを何れかの軸に平行なグループで一度にフェッチすると、隣接するグループ０およびグループ１などのグループは常に異なるバンクにあることが理解されるであろう。これは、４つのミニブロックのグループが、ＤＲＡＭチップの「バーストモード：burst mode」アクセスを利用して、ＤＲＡＭチップの部分にアイドルサイクルを介入させずに、またどの軸に沿って横断しても素早く連続してフェッチすることを意味する。また、これはＤＲＡＭ帯域幅の効率を最大化する。

【００５８】より一般的には、ミニブロックのメモリバンクへの割り当ては、ミニブロックのメモリチップへの割り当てと類似の方法でスキューできる。換言すると、ボリュームデータ集合がどの方向に横断されていても同時アクセスが可能なように、ミニブロックはＭ個のチップ全体に渡ってスキューすることができる。同様に、各チップ内のミニブロックはＢ個のメモリバンク全体に渡ってスキューすることができ、従って、バンク内の連続したミニブロックへのアクセスは、アイドルサイクルの介入によって遅延することはない。これは、チップとバンク全体に渡るミニブロックの２層のスキューイングを形成する。図示した実施の形態においては、ミニブロックのメモリバンクへの割り当ては次の公式によって求める。

$$Bank\ Number = [\,[X_{mb}/M] + [Y_{mb}/M] + Z_{mb}/M]\,] \bmod B \quad (6)$$

しかし、例えば３つの次元における距離が互いに相対的な基本となるように、異なる距離により各次元でスキューするというように、他の実施の形態では他の規則にしたがって、バンク全体に渡ってミニブロックをスキューしてもよいことが理解されるであろう。

【００５９】図示した実施の形態においては、ミニブロックは次のようにそのＤＲＡＭ内の特定のメモリアドレスに割り当てられる。ボリュームデータ集合が、Ｘ軸に沿った幅でＳＸミニブロック、Ｙ軸に沿った高さでＳＹミニブロック、Ｚ軸に沿った深さでＳＺミニブロックであると仮定する。ここで、これは合計ＳＸｘＳＹｘＳＺ個のミニブロックからなる。さらに、ＳＸ、ＳＹおよびＳＺのそれぞれが、ＤＲＡＭチップの数にチップ毎のバンクの数を掛けた倍数であると仮定する。すなわち、図示した実施の形態においては、ＳＸ、ＳＹおよびＳＺのそれぞれは１６の倍数である。次に、座標（X_{mb}、Y_{mb}、Z_{mb}）を有するミニブロックのMini Block Numberを次のように定義する。

$$Mini\ Block\ Number = X_{mb} + Y_{mb} \times SX + Z_{mb} \times SX \times SY \quad (7)$$

このＤＲＡＭ NumberおよびBank Numberへの割り当ては、式（３）および（５）によってそれぞれ求める。次に、そのバンク内のミニブロックのインデックス、Mini Block Index In Bankは次の式によって定義される。

$$Mini\ Block\ Index\ In\ Bank = [Mini\ Block\ Number/16] \quad (8)$$

すなわち、４つのＤＲＡＭモジュールがあり、それぞれに４つのバンクがあり、合計で１６個のバンクがある。従って、バンク内のミニブロックの位置は、そのMini-block Numberをバンクの総数で割ったものである。最後に、図示した実施の形態においては、バンクの各列は２個のミニブロック、すなわちそれぞれ１６ビットの３２×８個のボクセル、つまり合計４０９６ビットを保持することができる。従って、列内の列番号とオフセットは次の式によって求められる。

$$row = [Mini\ Block\ Index\ In\ Bank/32] \quad (9)$$

$$offset\ In\ Row = Mini\ Block\ Index\ In\ Bank \bmod 32 \quad (10)$$

この発明の趣旨の範囲内の他の公式によって、他の実施の形態でミニブロックのメモリバンク内のアドレスへの割り当てを行ってもよいことが理解されるであろう。

【００６０】横断順序：Traversal Order

この発明においては、観察方向に平行な光線が、メモリのボリュームデータ集合の方向には関係なく、ボリュームデータ集合を前から後ろ、左から右、上から下の方向に常に通過するように、ボリュームデータ集合はレンダリングの前に再方向付けされる。これは、ボリュームデータ集合のボクセルの横断を定義する。詳細には、後に処理されるボクセルから得られるサンプルを、先に処理されたボクセルのサンプルの後に合成できるように、横断は光線の方向に従わなければならない。従って、ボクセルの横断は、観察方向によってはボリュームデータ集

合の任意の角で始まってもよく、同様に観察方向によっては何れかの軸に沿ったＭ個のミニブロックのグループで進行してもよい。

【００６１】上記のバンクスキューイングの方法は、観察方向に拘わらず、何れの列内においても同一チップの同一バンクから連続的に２つのミニブロックをフェッチする状況を回避しているが、一般的に１つまたは２つの例外となる事態がある。これらは、ミニブロックの１つの列の終端からミニブロックの次の列の始点に横断することから生ずる。例を図８に示す。Ｘ軸に平行またはＺ軸に平行なグループでミニブロックをフェッチする場合には例外はないが、Ｙ軸に平行なグループについては例外がある。これは、原点がボリュームデータ集合の右上角になければならず、横断がまず正のＹ方向に進行し、次に負のＸ方向に、最後に正のＺ方向に進行しなければならないような観察方向の場合に発生する。すなわち、ボリュームデータ集合を横断する主たる方向は図８の破線３００に平行である。横断はグループ１、２、３および０のミニブロックを通って下方に進み、そのパスに沿って個々のボクセルを読み取る。この横断順序におけるミニブロックの次の列を破線３０２で示すが、この次の列も図の上から下に進行する。破線３００に沿った横断はバンク０で終わり、破線３０２に沿った横断は同じバンクで始まることが図から分かる。これはもちろん、禁止された一連のアクセスであり、パイプラインに挿入される余分なアイドルサイクルか、ＤＲＡＭチップのアクセス使用違反につながる。しかし、横断は破線３０２から破線３０４に進むので、例外はないことに注意されたい。

【００６２】例外的な事態に対する解決策は、横断の間にそれらを見つけて、ボリュームデータ集合の違反している列を、その前の列の反対方向に横断させることである。これを行うメカニズムを以下で説明する。

【００６３】デスキューイング：De-skewing

各ミニブロックは、そのメモリチップおよびバンクから１組の連続的なメモリアドレスとして読み取られる。従って、ミニブロックからボクセル値を読み取る順序は、必ずしもボクセルが処理される順序に対応しないことが理解されよう。この状況を考慮するため、デスキューイングの方法を以下に紹介する。

【００６４】ここで図９を参照すると、Ｍ個のミニブロック群からなるボクセルデータ値を再配置して、それらを正しい横断順序でボリュームレンダリングシステムの並行処理パイプラインに提供するための、デスキューイング回路を示してある。図９の上部では、Ｍ個の独立したＤＲＡＭチップ２３０が図５のボクセルメモリ１００から構成されている。ミニブロックはアドレスジェネレータ１０２の制御下でこれらＭ個のチップから同時に読み取られる。アドレスジェネレータ１０２はボリュームデータ集合の横断の順序で、ミニブロックのメモリアドレス２３４を生成する。ＤＲＡＭチップ２３０からのメモリ入力は、選択信号２３８を介してやはりアドレスジェネレータ１０２の制御下で機能する１組の選択ユニット２３６に連結されている。Ｍ個のミニブロックはそれが対応するメモリモジュール２３０から読み取られるので、選択ユニット２３６が、それらがどのメモリモジュールから来たかに拘わらず、その左から右への整列がボリュームデータ集合におけるミニブロックの物理的位置に対応するように、それらを効率的に再配置または並べ換える。すなわち、各選択ユニット２３６は、ＤＲＡＭチップの多くても１つからのその入力を選択し、各ＤＲＡＭチップ２３０は多くても１つの選択ユニットによって選択される。

【００６５】選択ユニット２３６の出力は次に、ミニブロックデスキューイングユニット２４０に連結される。信号線２４２を介してアドレスジェネレータ１０２の制御下で機能しているので、各ミニブロックデスキューイングユニットは、データ値が横断の順序に関連した各ボクセルの物理的位置に対応する順序、例えばその自然な順序で提供されるように、そのミニブロック内でデータ値を再配置する。ボクセル値の合計Ｐ個のストリームは、ミニブロックデスキューイングユニットから出力され、図５に示された種類のＰ個のパイプラインの補間（内挿）ユニット１０３に連結される。メモリチップの数Ｍは、処理パイプラインＰの数よりも少なくても、それと同じでも、それより多くてもよいことが理解される。

【００６６】上記の手段によって、メモリやバンクのコンフリクトによる遅延はなく、何れの観察方向からも１サイクル当たり１つのボクセルデータ値の一定したレートで、ボクセルメモリからデータを読み取ることが可能であるが、例外が１つある。その例外は上述したように、１つの行または列の終端のバンクが他の行または列の始点のバンクと同じ場合である。この例外が見つけられなければ、各ＤＲＡＭチップは同じバンクから連続的な二番目のミニブロックを読み取るためにそのバンクをプリチャージする一方で、違反している列の終端では数サイクルの遅延が生じることになる。この遅延は図５のパイプライン全体に伝わり、余分な制御回路や複雑性を伴うこととなる。この問題を緩和するためには、図１０に示したように、追加のバッファ２４４をＤＲＡＭチップ２３０と選択ユニット２３６との間に設ける。各バッファは、１つの列において単一のメモリモジュール２３０から読み取られる分のミニブロックを収容するには十分大きい。違反した列に遭遇すれば常に、ＤＲＡＭチップからのフェッチングが、正常な上から下または左から右への方向ではなく、下から上または右から左の方向に行われ、また、アドレスジェネレータ１０２の制御下で進む。ミニブロックの列に関するデータはバッファに記憶され、行または列がそれぞれ正常な行または列である

か、違反している行または列であるかによって、先入れ先出しの順序または後入れ先出しの順序で選択ユニット２３６によって取り出される。

【００６７】図示した実施の形態においては、ボリュームデータ集合はセクションに分割されている。通常のセクションは横断方向の各列に３２ボクセルまたは１６ミニブロックを有する。メモリモジュールの数Mは４である。従って、各バッファ２４４は、列を何れかの順序で横断することができるように、４つのミニブロックのみを保持する必要がある。この手段によって、ボクセルデータは、観察方向に拘わらず、中断やアイドルサイクルの介入や遅延なく、ＤＲＡＭメモリモジュールの最大定格速度のバーストで読み取ることができる。

【００６８】以上、この発明のいくつかの実施の形態、並びにそれに対する修正や変更を説明してきたが、当業者には上記が単に解説的なものであって限定的ではなく、例示のみによって示されていることが明らかなはずである。多くの修正や他の実施の形態が当該技術分野の範囲にあり、請求項およびその均等物によって定義された発明の範囲に入るものと予想する。

【００６９】

【発明の効果】以上のように、この発明によれば、１組のＤＲＡＭメモリモジュールに割り当てられたミニブロック内にボクセルデータが記憶されたボリュームレンダリングシステムのメモリサブシステム上に２層のスキューイングアーキテクチャを組み付け、それによって、ＤＲＡＭメモリの最大バーストレートでのデータ転送を可能にし、実時間ボリュームレンダリングを可能にすることができる。

【図面の簡単な説明】

【図１】　ボリュームデータ集合の線図である。

【図２】　光線キャスティングによって画像平面に投影されたボリュームデータ集合の画像の線図である。

【図３】　図２のボリュームデータ集合の断面図である。

【図４】　光線キャスティングによる個々の光線の処理の線図である。

【図５】　この発明による実時間ボリュームレンダリングのためのパイプライン化された処理エレメントのブロック図である。

【図６】　ＳＤＲＡＭへのミニブロックからなるボクセルのマッピングの線図である。

【図７】　メモリ内のミニブロックの線図である。

【図８】　ＤＲＡＭのバンクと列の中のミニブロックの線図である。

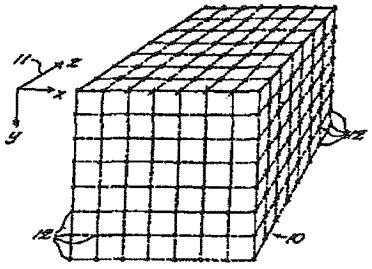
【図９】　ミニブロックのボクセルを自然な処理順序に再配置するためのデスキューイング回路のブロック図である。

【図１０】　ミニブロックの列を読み取って何れかの順序で進行することを可能にする目的で、図９へのバッファの付加を示すブロック図である。

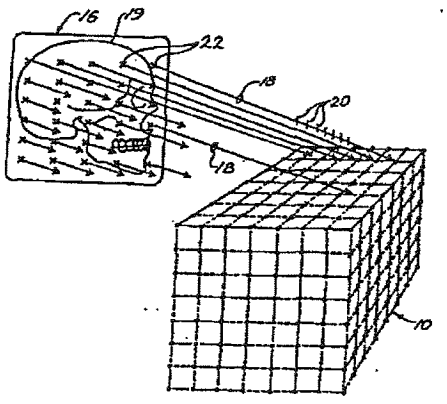
【符号の説明】

１０　ボリュームデータ集合、１１　軸、１２　ボクセル　ボクセル位置、１６　二次元画像平面、１８　光線、１９　画像、２０　サンプルポイント、２２　ピクセル　ピクセル位置、２５　勾配、３２　画像、１００　ボクセルメモリ、１１０　ＦＩＦＯエレメント、１１１　勾配推測ユニット、２００　ミニブロック、２０２　ボクセル、２０４　３ビット二進数、２０６　軸、２０８メモリ、２０９　方向、２１１　方向、２１２　ミニブロック、２１４　グループ、２３０　ＤＲＡＭチップ、２３４　メモリアドレス、２３６　選択ユニット、２３８　選択信号、２４０　ミニブロックデスキューイングユニット、２４２線、２４４　バッファ、３００　破線、３０２　破線、３０４　破線。

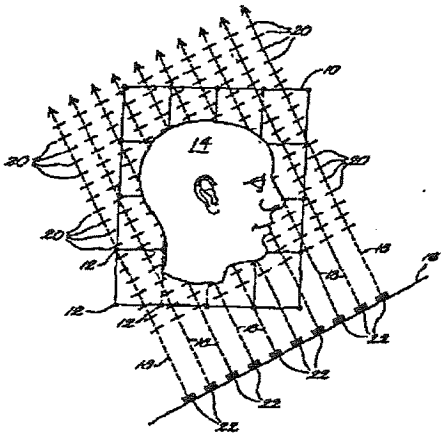
【図１】



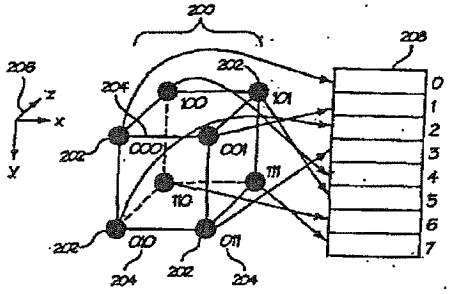
【図２】



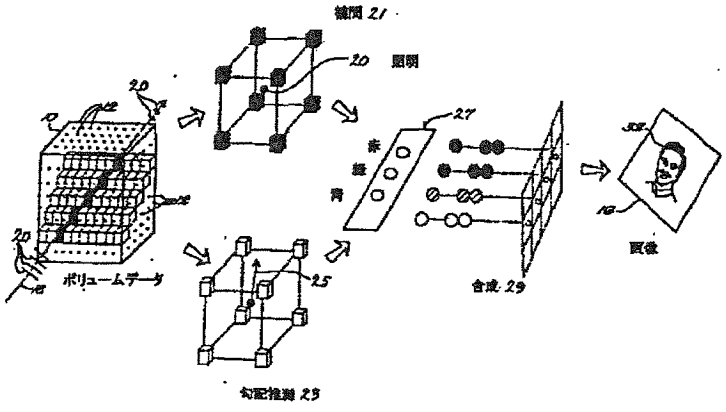
【図3】



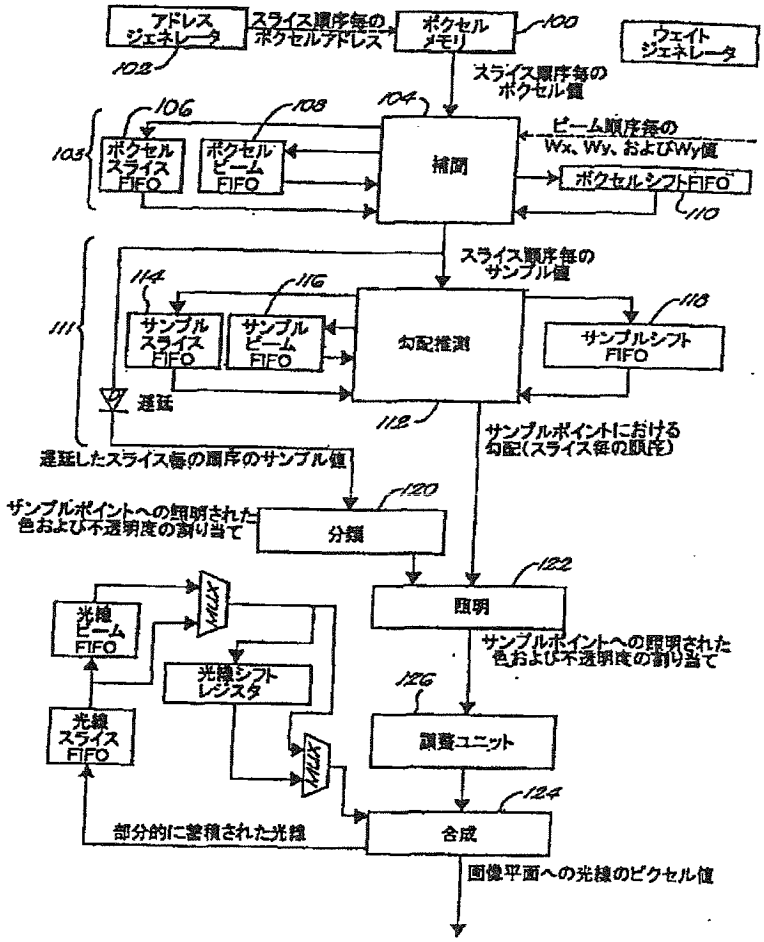
【図6】



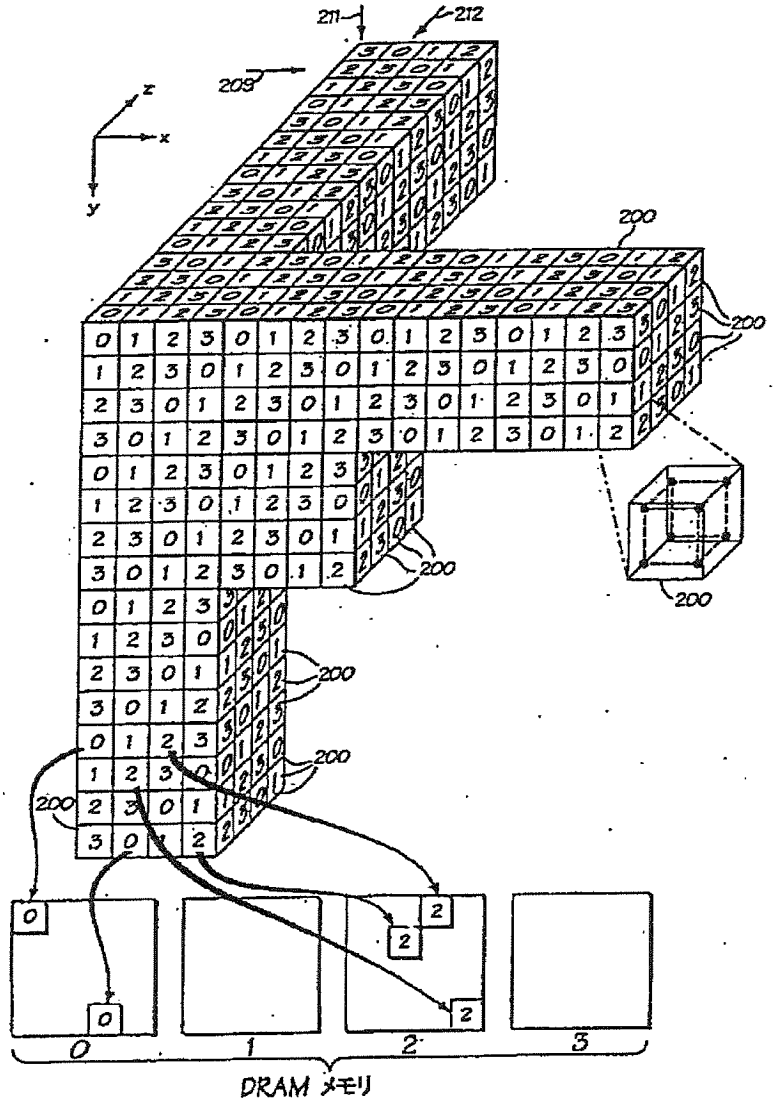
【図4】



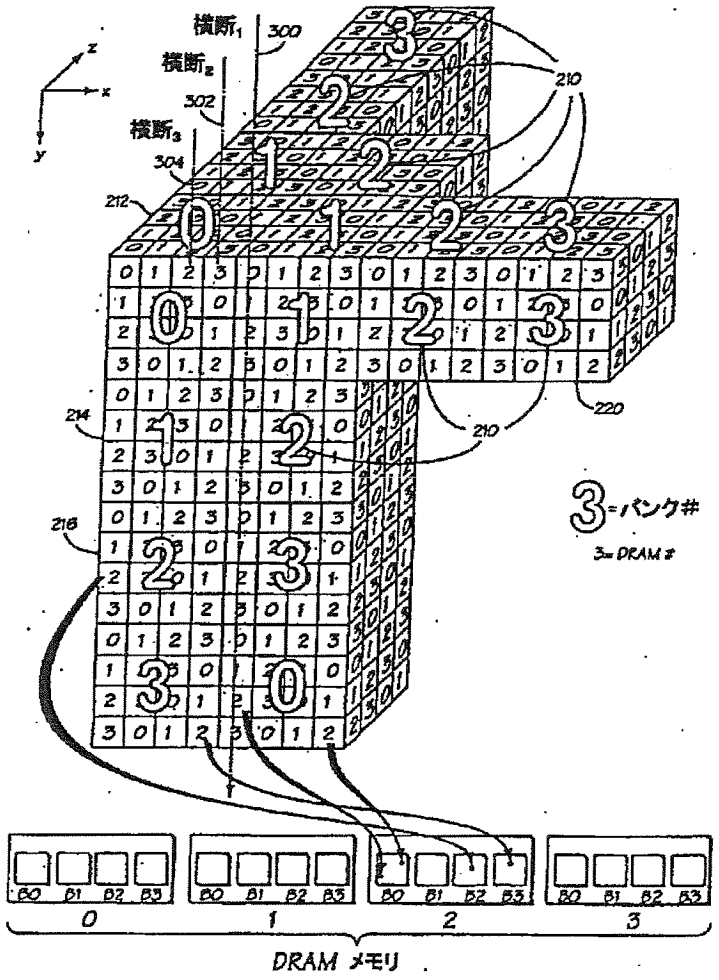
【図５】


アドレス ジェネレータ
スライス順序毎の ピクセルアドレス
ピクセル メモリ
100
ウェイト ジェネレータ
102
スライス順序毎の ピクセル値
104
106
108
103
ピクセル スライス FIFO
ピクセル ビーム FIFO
補間
ビーム順序毎の Wx、Wy、およびWy値
ピクセルシフトFIFO
110
スライス順序毎の サンプル値
114
116
111
サンプル スライス FIFO
サンプル ビーム FIFO
勾配推測
118
サンプルシフト FIFO
遅延
112
サンプルポイントにおける 勾配(スライス毎の順序)
遅延したスライス毎の順序のサンプル値
サンプルポイントへの照明された 色および不透明度の割り当て
120
分類
122
照明
MUX
光線 ビーム FIFO
光線シフト レジスタ
サンプルポイントへの照明された 色および不透明度の割り当て
126
光線 スライス FIFO
MUX
調整ユニット
124
合成
部分的に蓄積された光線
画像平面への光線のピクセル値

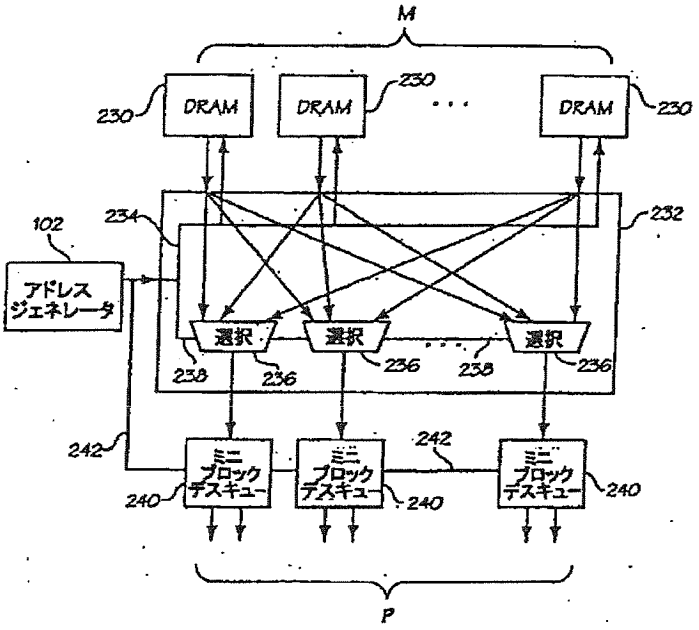
【図７】



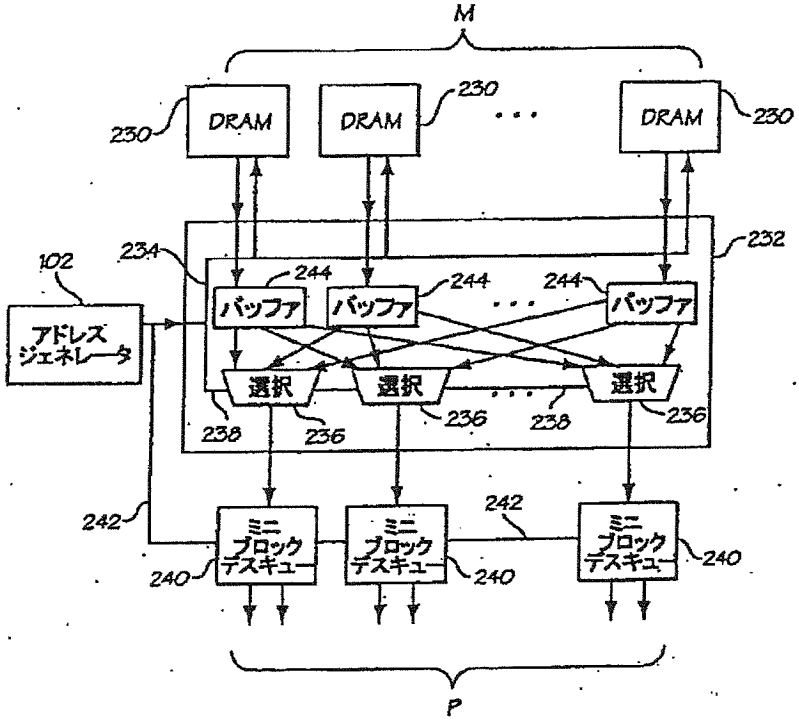
【図8】


横断1
300
横断2
302
横断3
304
210
212
214
216
220
3=バンク#
3=DRAM #
B0 B1 B2 B3
0
1
2
3
DRAM メモリ

【図９】


M
DRAM
DRAM
DRAM
230
102
234
232
アドレス
ジェネレータ
選択
選択
選択
238
236
242
ミニ
ブロック
デスキュー
240
P

【図１０】


M
DRAM
DRAM
DRAM
230
102
234
232
244
バッファ
アドレス
ジェネレータ
選択
238
236
242
ミニ
ブロック
デスキュー
240
P

フロントページの続き

(71)出願人 597067574
201 BROADWAY, CAMBRIDGE, MASSACHUSETTS 02139, U. S. A.
(72)発明者 ジェームズ・ニッテル
アメリカ合衆国、マサチューセッツ州、グラトン、ヒル・ロード 241
(72)発明者 ウイリアム・ピート
アメリカ合衆国、マサチューセッツ州、ピレライカ、メリディエン・ウェイ 6
(72)発明者 ケネス・コレル
アメリカ合衆国、マサチューセッツ州、ランカスター、ルネンバーグ・ロード 2193